A Study of a Server Selection Model for Selecting a Replicated Server based on Downstream Measurement in the Server-side

Seung-Hae Kim, Won-Hyuk Lee, and Gi-Hwan Cho

Abstract: In the distributed replicating server model, the provision of replicated services will improve the performance of the providing service and efficiency for clients. Efficiently composing the server selection algorithm decreases the retrieval time for replicated data. In this paper, we define the system model that selects and connects the replicated server that provides an optimal service using the server-side downstream measurement and propose a server selection algorithm.

Keywords: server selection, replicated server, measurement

1. Introduction

Nowadays, file transmission and receipt via the internet has become the most common means of study, work, and leisure all over the world. So the action of downloading files or receiving a web service using the internet is often performed. The extension of the internet's service scope has made it available all over the world. Therefore, users who want to access the same service are distributed all over the world, so a server system can be constructed that is distributed by several servers and installed in many regions. [1, 2]

It is necessary to connect the propriety server in a distributed server system to provide the service desired a client, and these methods have been studied. A method is used that simply connects the near server, although it is not correct all the time. So it is important that each user chooses the server that is suitable to the user's desired location, as many mechanisms that perform server selection are used.

The server selection algorithm that is efficiently organized decreases the search time for data that is replicated by other servers.

In the existing model, it was performed to select a server by client, as being based on it, client was served wanted service. However, this method may cause a delay in serving the required service to the client and such delays are harmful to performance.

So, in this paper, we have proposed a system model that provides a suitable service on the server side downstream measurement, and studied the proper algorithm. Finally, we compared the existing algorithm on the client side measurement with the proposed algorithm and we verified the results. [5], [6], [7]

Section II presents the system model for selecting a replicated server proposed in this paper, and section III compares the proposed model with the existing model using a test. Finally, section IV observes the significance of this model and summarizes.

Manuscript received September 29, 2005; accepted May 22, 2006
Corresponding Author: Seung-Hae Kim
* KISTI and Chonbuk University, Korea (shkim@kisti.re.kr, livezone@kisti.re.kr, ghcho@dcs.chonbuk.ac.kr)

2. Proposal of a selection model for a replicated server

In this paper, we propose a server selection model with which it is possible to select and serve the optimal service for clients from among the distributed servers

This consists not only of performing a performance test on each server by client, but could also represent an optimal selection to serve a service because it tests performance from the viewpoint of the server.

2.1 Pre-defined notation

Def 1) Define the total number of servers serving a service in the directory service

$$T = \{ t1, t2, \ldots, tk \} \quad \text{(k is the total number of service servers)} \tag{1}$$

Def 2) Define the set of servers that could provide a service wanted by the client.

$$R = \{ r1, r2, \ldots, rk \} \quad \text{(k is the total number of replicated servers)} \tag{2}$$

Def 3) Define the set of servers that does not belong to the replicated servers among the service servers of the total set.

$$N = T - R = \{ n1, n2, \ldots, nk \} \tag{3}$$

Namely, total set T, R and N has the following properties.

$$T = R \cup N, \quad R^c \cap N^c = \phi \tag{4}$$

Def 4) The round trip time that sends and receives messages from the client to the specific server is defined as the following:

$\delta(r_k)$: The transmission time from client to r_k node

$\vartheta(rk)$: The transmission time from r_k node to client (5)

Def 5) The following is the total time taken to receive a response after the client sends a message to each server:

$$\xi(r_k) = \delta(r_k) + \vartheta(r_k) \tag{6}$$

In this rule, we send the messages from the client to each server rk, and select the least value received in (*rk*), and we define it as the least time of measurement, MIN_Svr_Time.

$$MIN\_Svr\_Time = Min(\xi(r_1), \xi(r_2), \ldots, \xi(r_k)) \tag{7}$$

Def 6) The server *rk* selected by MIN_Svr_Time is the server that could serve a service wanted by client in the shortest time.

3. Experimentation

3.1 Simulation and development environment

In order to conduct the experiment, we constructed an environment for the simulation. We wrote the simulator in visual C++ and ran it on a Windows system that could support Windows Library and API.

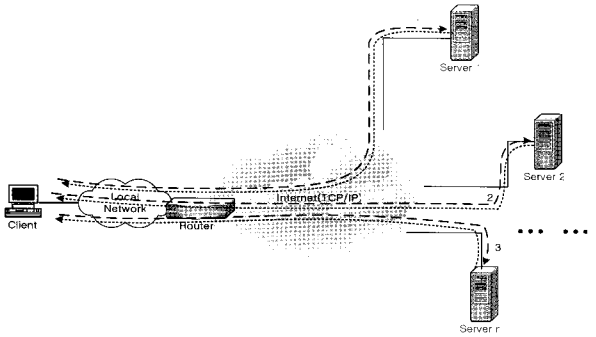


Fig. 1. The method and step for the downstream measurement in the server-side after request by client.

As shown in the figure 1, we send a message simultaneously from the client to each server to request a measurement of server selection. Then, we select the fastest server by measuring the downstream performance from among the response messages of each server.

3.2 Comparison of the existing model and this model

a. The problem with the existing model

1. If a delay from the host to the server that serves the service occurs because of overload on the current network, even though it is a temporary phenomenon, it can not perform the appropriate server selection. [10]
2. It is reported that the path from source to destination may be different from the path from destination to source because of the asymmetric path of IP networks in RFC 2679. And, though the path is symmetric in structure, it is clearly stipulated in that paper that it appears as a different performance characteristic because of asymmetric queuing.

b. The characteristics of using a downstream measurement

1. Client sends requests *n* times to each server in parallel.
2. Each server that received a request from the client resends the measurement data to the client using a downstream measurement.
3. The client selects the appropriate server that will serve the service based on the speed of the response coming from each server.

3.3 Choosing a replicated server based on the downstream measurement

The path could be altered when the client requests the service and then receives the real service. On the assumption that there is a replicated server *1* like (a) of fig. 2, the path by which that client requests the service is established as path2, but it could be established as path1 when the server serves the real service data.

Let us suppose that the paths by which the client requests and receives the service are path2 and path1.

When the client sends a test message for selecting a server, we suppose that the paths of replicated server 1 and server 2 are established as (a) and (b) of fig. 2, and T1, T2, the total times of server1 and server2, are taken as the following times:

$$\begin{aligned} T1 &= T1path1 + T1path2 \\ T2 &= T2path1 + T2path2 \end{aligned} \tag{8}$$

In the case of T1, the request time increases because of the detour made for path2. The response time of T2 may be faster than T1's response time. However, although T1path1, the time that served the real services, may be faster than T2path1, we do not know its difference using the existing

method.

That is, in the existing method, it only considers the round trip time from the client to the server, and selects the fastest server for a reply time, but in the algorithm of this paper, we select the server for measuring a downstream time at the server that serves the real service even though it has a bad round trip time. Therefore, when using this algorithm we can select an appropriate server for the situation of receiving real data.

Furthermore, we suppose that the time taken to request a test from a specific client to server k is Tkpat1, and the time taken to respond by server k to the client is Tkpath2, while the time *Tk* that it takes to select a server for the client is as follows:

$$\begin{aligned} T1 &= T1path1 + T1path2 \\ T2 &= T2path1 + T2path2 \end{aligned} \quad (9)$$

Therefore, the total time T for measuring at all servers, is as follows:

$$T = \sum_{k=1}^{n} Tk_{path1} + \min (T1_{path2}, T2_{path2}, \ldots., Tn_{path2}) \quad (10)$$

Therefore, in this model which measures a downstream performance on the server side, we do not select a server based simply on RTT on the client side, but measure a downstream that will serve the real service based on the time of the request time and response time, so we are able to select the appropriate server accurately using this algorithm. So, the total time T is as follows:

$$T = \sum_{k=1}^{n} Tk_{path1} + \sum_{k=1}^{m} \min (T1_{path2}, T2_{path2}, \ldots., Tk_{path2}) \quad (11)$$

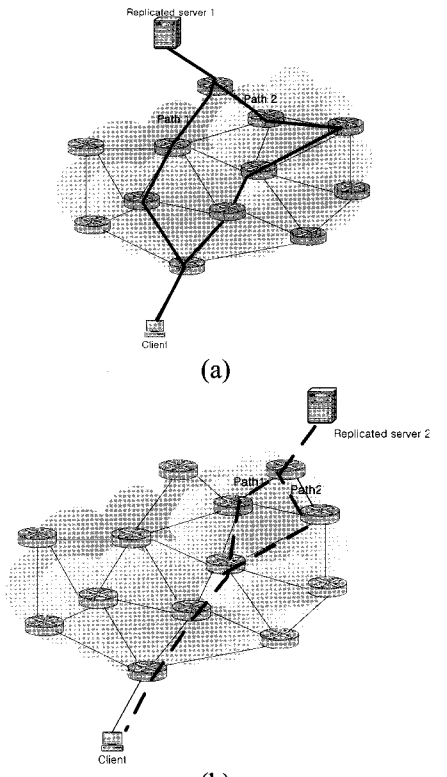


Fig. 2. figure (a) presents the request and served path of the service at the replicated server 1, figure (b) presents the request and served path of the service at replicated server 2

3.4 Simulation of the proposed downstream measurement model

Firstly, in order to determine the relation of the file size and response time, we select three servers which satisfy the following conditions and conduct an experiment to determine the relation between the number of hops and the response time.

Conditions)

1. Server A is located in Daejeon, Korea. From the test area to server A consists of 7 hops.
2. Server B is located in Japan. From the test area to server B consists of 12 hops
3. Server C is located in the USA. From the test area to server C consists of 12 hops.

We measure the response time as increasing a size of files, for servers A, B, C which satisfy the above conditions; Table 1 presents the results.

Table 1. The variation of the response time according to the increase in file size

	File size (bytes)				
Server	1,024	2,048	10,240	20,480	30,720
Server A	9.79	12.726	34.104	46.498	63.999
Server B	48.256	48.867	58.033	65.797	77.623
Server C	132.817	133.695	138.272	142.677	147.253

Also, fig. 3 presents the change in response time for each server as the file size increases. We determined that the response time changes as the file changes in size, but it does not increase linearly, as shown in this figure.

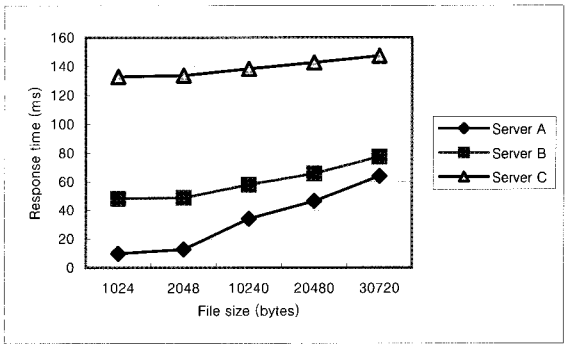


Fig. 3. Relationship between file size and response time in the test server. This shows an increase in the response time according to an increase in files size.

Though the hops of servers B and C are equal to 12, this differs from response time, and the rate of increase differs according to the increase in file size. Therefore, to select a server using only hops isn't suitable in reality. Also, the rate of increase does not increase linearly as the file changes in size.

Though we do not consider a path MTU of the intermediate network, it is of no significance to consider a change of response time as changing a file size because of not exceeding 65535 bytes for IP packet.

So, in this paper, we use messages of 2~3kbytes as a test message, and compare our proposal algorithm with the existing method in order to select an appropriate server and connect it.

If we define the required time of upstream traffic from the specific client to the server as USt, and define the required time of downstream traffic as DSt , then we can divide a server type as shown in fig. 4. The number of upstream and downstream presents a weight.

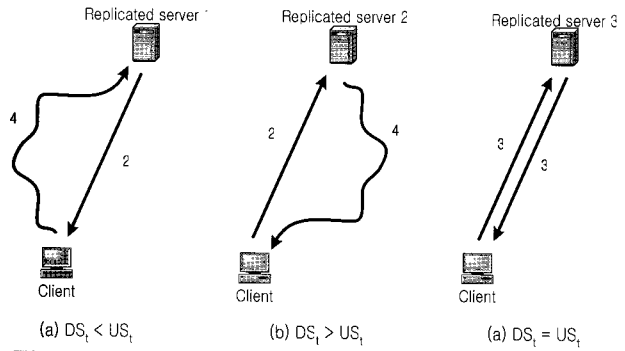


Fig. 4. Server selection model according to DSt and USt

a. $\frac{DS_t}{US_t} > 1$

This is the case where we select the server using only the calculation of RTT, as with the existing method. This type is the same case as (b) of fig. 4. Judging from the served point of view, as the service starts, it may induce a terrible performance.

b. $\frac{DS_t}{US_t} = 1$

This case is the same as the existing model where USt = DSt = RTT. In this case, the same performance may occur whether we use this model or the existing model.

c. $\frac{DS_t}{US_t} < 1$

Like (a) of fig. 4, this is the best case because a good service time is achieved and the shortest path is taken although it takes more time.

Fig.5 presents the results of the time taken by replicated servers to perform the relation test for upstream and downstream.

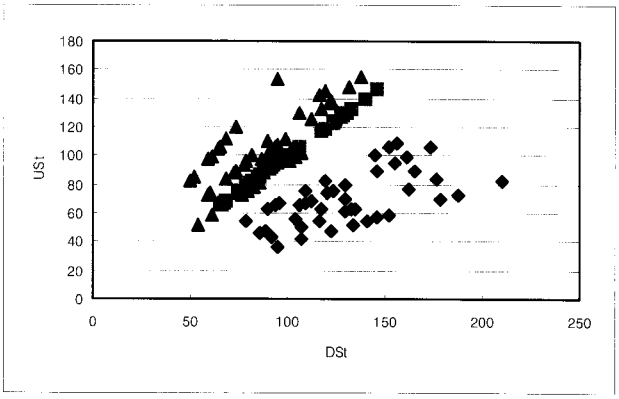


Fig. 5. The measurement time of DSt and USt

In figure 5, the mark (■) fits the (c) pattern of figure 4 on occasions where US and DS take the same time. This value is seldom appeared character in practical experiments, but we set up a border just to compare (a) with (b).

The case of figure 4, (a) is the occasion where the values DSt / USt are smaller than 1, and we select a server from this area, where the symbol (▲) presents an optimal service when any client is provided with practical services by an optimal server.

4. Conclusion

In this paper, we suggest a model for selecting a server that can provide an optimal service to the client desiring a service from among replicated servers. Thus, in order to provide mass and high-speed services, we suggest a model based on a new system when we perform server selection in an environment extending from a single server to numerous servers.

In the existing method, the system carried out server selection by measuring RTT after considering many capacity factors on the client side. However, because of the characteristics of the IP network, the path that it requests and responds to may differ by network load or because of temporary confusion. In such a case, the course requested will take more time than the response, although the contrary may also be the case.

On this occasion, a downstream measurement which considers the traffic direction of the server that can serve the real service is essential in order to select a server more correctly.

When a server is selected and offered, such a one-side measurement is suitable in reality because its efficiency is determined by downstream on the server part.

Furthermore, in the existing model, the system has the precondition that the list of servers remains static, but in this model, we use the directory service dynamically to renew location information, so it provides a model that can

use the newest information.

In this paper, we suggest a method for server selection that is able to offer the most appropriate performance via a one-side measurement from the server, excepting the case of downstream detours, from among several server types as previously suggested. As such, this case showed superior efficiency.

At present, P2P (Peer-to-Peer) is the dominant trend. The various solutions and application programs used this appears in sight. Replicated servers of this paper can apply after extending from research model about to P2P, if replicated servers are considered each server. Gradually, as the capacity of personal computers and computing power grows, each PC will do a roll of server enough. Therefore, studies on applicable repeated server widely will be required.

References

[1] R.Tewari, M. Dahlin, H. M. Vin, and J. S. Kay, "Design considerations for distributed caching on the Internet", Proceeding of the Int'l. conference on Distributed Computing Systems(ICDS'99), 1999.

[2] L. Zhang, S.Floyd, and V.Jacobson, "Adaptive web caching", Proceeding of the 2nd Web Cache Workshop, 1997. 6.

[3] Sandra G. Dykes, Kay A. Robbins, Clinton L. Jeffery , "An Empirical Evaluation of Client-side Server Selection Algorithm", Proceedings of IEEE Infocom, Tel-Aviv, Israel, March 2000.

[4] Robert L. Carter, Mark Crovella, "Server Selection Using Dynamic Path Characterization in Wide-Area Networks", Proceedings IEEE INFOCOM '97, The Conference on Computer Communications, April 7-12, 1997.

[5] Jiahai Yang, Peiyu Wang, Jianping Wu, "A scalable, web-based architecture for hierarchical network management", Global Telecommunication Conference, 1999. GLOBECOM '99, Volume: 3, 1999, page 1882 -1888 vol.35.

[6] K. Ikehara, K. Ikeda, K. Motomura, "Proposal of a hierarchical network management method based on network management protocol monitoring", 2000 IEEE/IFIP Network Operations and Management Symposium, 2000.

[7] Adarshpal S.Sethi, "A Hierarchical Management Framework for Battlefield Network Management" , MILCOM 97 Proceedings, 1997.

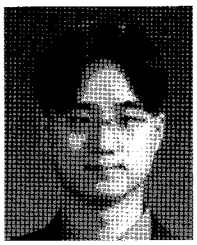
[8] Mehmet Sayal, Yuri Breitbart, Peter Scheeuermann, Redek Vingralek, "Selection Algorithm for Replicated Web Servers", Proceedings of the Workshop on Internet Server Performance, 1998.

[9] James Gwertzman and Margo Seltzer, "The case for geographical push-caching", Proc of the 1995 workshop on Hot Operating Systems (HotOS-V), 1995, pp51-55.

[10] G. Almes, S. Kalidindi, M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, september 1999.

Seung-Hae Kim

He received the BS in Information and Communication Engineering from Hannam Univ. in 1997 and MS degrees in Information Science from Chonbuk Univ. in 2003. During 1996~2006, he stayed in Korea Institute of Science and Technology Information to manage and analyze especially the Research Network. And now he is undertaking a doctorate course in Information & Security Engineering at Chonbuk National University. His research interests include network architecture and security.



Won-Hyuk Lee

He received the BS and MS degrees in Computer Engineering from SungKyun-Kwan Univ. in 2001 and 2003. During 2003~Present, he stayed in Korea Institute of Science and Technology Information to manage and analyze the network traffic. His research interests include network architecture and security.

Gi-Hwan Cho

Gihwan Cho received the B.Sc. degree in Computer Science and Statistics from Chunnam National University, S. Korea, in 1985; and M.Sc. degree in Computer Science and Statistics from Seoul National University, S. Korea in 1987; both major in Computer Science, and the Ph.D. degree in Computer Science from University of Newcastle, Newcastle upon Tyne, England, in 1996. From 1987 to 1997 he worked for ETRI, Daejeon, S. Korea. From Sep. 1997 to Feb. 1999, for the Dept. of Computer Science at Mokpo National University, Mokpo, S. Korea, as a full time lacture. From Mar. 1999, he joined to the Division of Electronic & Information Engineering at Chonbuk National University, Chonju, S. Korea, as an Professor.